Санкт-Петербургский национальный исследовательский университет
информационных технологий, механики и оптики

Факультет инфокоммуникационных технологий

Направление подготовки 11.03.02

Лабораторная работа №5

Создание и использование массив

Выполнил:

Дощенников Никита Андреевич

Группа: К3221

Проверил:

Иванов Сергей Евгеньевич

Санкт-Петербург

2025

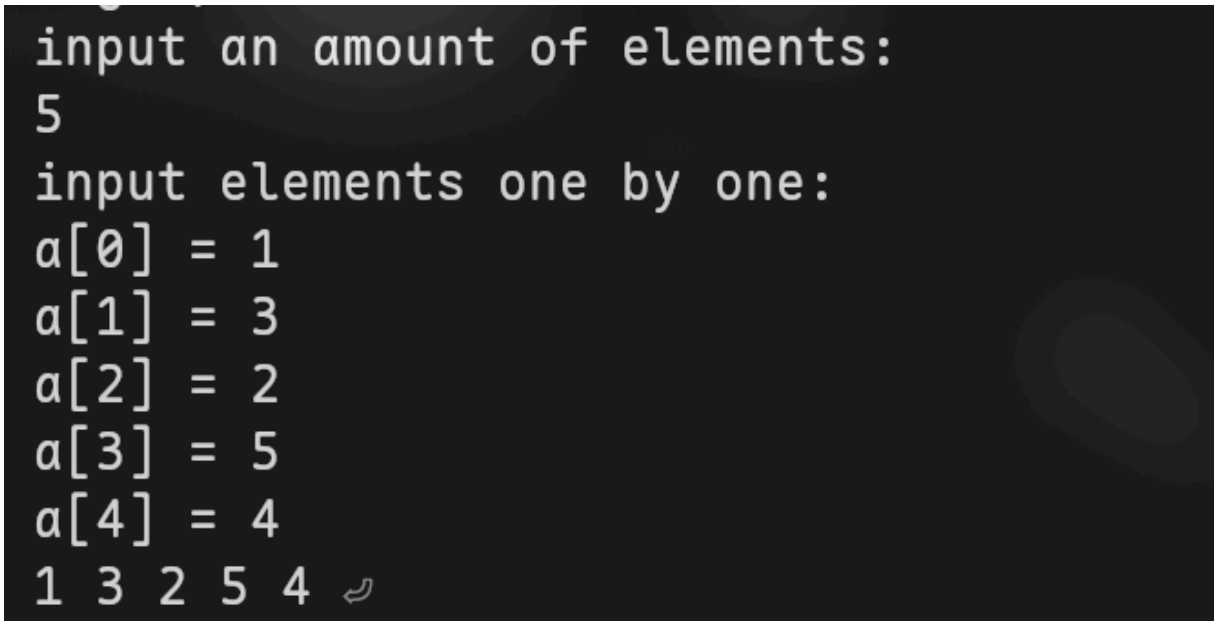
Цель работы:
Изучить массивы и приобрести навыки работы с ними.

Упражнение 1. Работа с массивом размерного типа данных.
В этом упражнении в проекте `Loop` (из третьей лабораторной) я реализовал массив для хранения данных.

```
public static void Main(string[] args)
{
    int[] myArray = {100, 1, 32, 3, 14, 25, 6, 17, 8, 99};
    int n = 10;
    for (int i = 0; i < n; i++)
    {
        if (myArray[i] % 2 == 0) myArray[i] = 0;

        Console.Write(myArray[i] + " ");
    }
}
```

Примеры:


```
public static void Main(string[] args)
{
    int[] myArray;
    Console.WriteLine("input an amount of elements: ");
    int n = int.Parse(Console.ReadLine());
    Console.WriteLine("input elements one by one: ");
    myArray = new int[n];

    for (int i = 0; i < myArray.Length; ++i)
    {
        Console.Write("a[{0}] = ", i);
        myArray[i] = int.Parse(Console.ReadLine());
    }

    foreach (int x in myArray) Console.Write("{0} ", x);
}
```



Упражнение 2. Перемножение матриц.

В этом упражнении я написал программу перемножения матриц 2 на 2.

```
using System;

public class Program
{
    public static void Main(string[] args)
    {
        int[,] m1 = new int[2, 2];
        int[,] m2 = new int[2, 2];

        fillQuadraticMatrix(ref m1, 2);
        fillQuadraticMatrix(ref m2, 2);

        int[,] res = multiplyQuadraticMatrix(m1, m2, 2);

        printQuadraticMatrix(res, 2);
    }

    static void fillQuadraticMatrix(ref int[,] m, int n)
    {
        for (int i = 0; i < n; i++)
        {
            for (int j = 0; j < n; j++)
            {
```

```
                Console.Write("m[{0}][{1}] = ", i, j);
                m[i, j] = int.Parse(Console.ReadLine());
            }
        }

    }

    static void printQuadraticMatrix(int[,] m, int n)
    {
        for (int i = 0; i < n; i++)
        {
            for (int j = 0; j < n; j++)
            {
                Console.Write(m[i, j] + " ");
            }
            Console.Write("\n");
        }
    }

    static int[,] multiplyQuadraticMatrix(int[,] m1, int[,]
m2, int n)
    {
        int[,] res = new int[n, n];

        for (int i = 0; i < n; i++)
        {
            for (int j = 0; j < n; j++)
            {
                for (int k = 0; k < n; k++)
                {
                    res[i, j] += m1[i, k] * m2[k, j];
                }
            }
        }
        return res;
    }
}
```

Тесты:

Упражнение 3. Обработка данных массива.

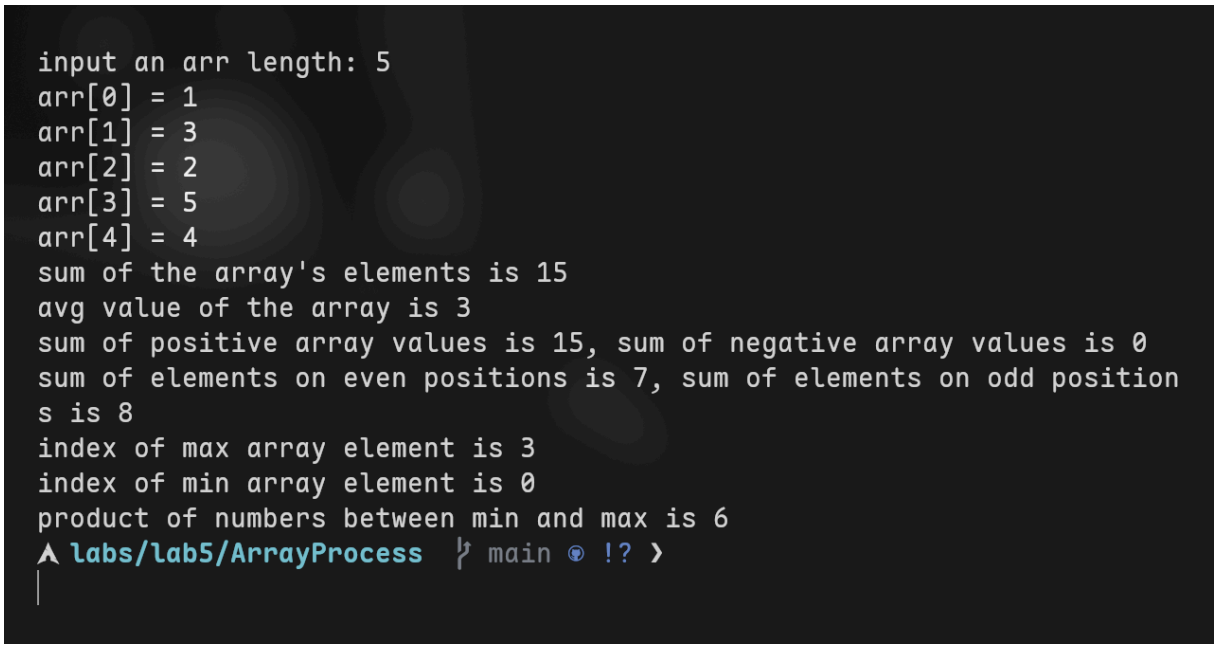
В этом упражнении я написал программу, которая заполняет массив элементами пользователя и выполняет обработку данных, а именно определяет сумму всех элементов массива, среднего значения массива, расчет суммы отрицательных или положительных элементов, расчет суммы элементов с нечетными или четными номерами, находит максимальный или минимальный элемент массива и вывести их индексы, расчитать произведение элементов массива, расположенных между максимальным и минимальным элементами.

```
using System;

public class Program
{
    static int SumPos(int[] a)
    {
        int r = 0;
        foreach (int x in a) if (x > 0) r += x;
        return r;
    }

    static int SumNeg(int[] a)
    {
        int r = 0;
        foreach (int x in a) if (x < 0) r += x;
```

```
        return r;
    }

    static int SumArr(int[] a) => SumPos(a) + SumNeg(a);

    static double AvgArr(int[] a) => a.Length == 0 ?
double.NaN : (double)SumArr(a) / a.Length;

    static int SumEvenPos(int[] a)
    {
        int r = 0;
        for (int i = 0; i < a.Length; i += 2) r += a[i];
        return r;
    }

    static int SumOddPos(int[] a)
    {
        int r = 0;
        for (int i = 1; i < a.Length; i += 2) r += a[i];
        return r;
    }

    static int MaxIdx(int[] a)
    {
        if (a.Length == 0) return -1;
        int idx = 0, m = a[0];
        for (int i = 1; i < a.Length; i++)
            if (a[i] > m) { m = a[i]; idx = i; }
        return idx;
    }

    static int MinIdx(int[] a)
    {
        if (a.Length == 0) return -1;
        int idx = 0, m = a[0];
        for (int i = 1; i < a.Length; i++)
            if (a[i] < m) { m = a[i]; idx = i; }
        return idx;
    }

    static int ProductBetweenMinMax(int[] a)
    {
```

```
        int start = MinIdx(a), end = MaxIdx(a);
        if (start == -1 || end == -1) return -1;
        if (start > end) { int t = start; start = end; end =
t; }
        if (end - start <= 1) return -1;
        int r = 1;
        for (int i = start + 1; i < end; i++) r *= a[i];
        return r;
    }

    public static void Main(string[] args)
    {
        Console.Write("input an arr length: ");
        int n = int.Parse(Console.ReadLine());
        int[] arr = new int[n];

        for (int i = 0; i < arr.Length; i++)
        {
            Console.Write("arr[{0}] = ", i);
            arr[i] = int.Parse(Console.ReadLine());
        }

        Console.WriteLine("sum of the array's elements is
{0}", SumArr(arr));
        Console.WriteLine("avg value of the array is {0}",
AvgArr(arr));
        Console.WriteLine("sum of positive array values is
{0}, sum of negative array values is {1}", SumPos(arr),
SumNeg(arr));
        Console.WriteLine("sum of elements on even positions
is {0}, sum of elements on odd positions is {1}",
SumEvenPos(arr), SumOddPos(arr));
        Console.WriteLine("index of max array element is {0}",
MaxIdx(arr));
        Console.WriteLine("index of min array element is {0}",
MinIdx(arr));
        Console.WriteLine("product of numbers between min and
max is {0}", ProductBetweenMinMax(arr));
    }
}
```

Тесты:



Выводы.
В ходе проделанной работы я изучил массивы и приобрел навыки работы с ними.

Code review. (by zzzcode.ai)
Резюме

Код представляет собой набор функций для работы с массивами целых чисел, включая вычисление суммы положительных и отрицательных элементов, среднего значения, суммы элементов на четных и нечетных позициях, а также индексов максимального и минимального элементов. Также реализована функция для вычисления произведения элементов между минимальным и максимальным значениями.

Ошибка

Код не обрабатывает возможные исключения, такие как `FormatException` при вводе данных пользователем. Это может привести к сбоям программы, если пользователь введет некорректные данные.

Стиль кода

Стиль кода в целом соответствует стандартам C#. Однако, использование однострочных конструкций `if` может ухудшить читаемость. Рекомендуется использовать фигурные скобки для всех условных операторов.

Структура кода

Код организован в виде статических методов внутри класса `Program`. Это позволяет легко вызывать методы из `Main`. Однако, стоит рассмотреть возможность разделения логики на несколько классов для улучшения модульности.

Читаемость

Читаемость кода в целом хорошая, но можно улучшить, добавив комментарии к методам, чтобы объяснить их назначение. Также стоит использовать более описательные имена переменных.

Производительность

Производительность кода в целом приемлема для небольших массивов. Однако, для больших массивов стоит рассмотреть возможность использования параллельных вычислений или оптимизации алгоритмов.

Масштабируемость

Код может столкнуться с проблемами масштабируемости при работе с большими массивами, особенно в методах, которые используют вложенные циклы. Рекомендуется оптимизировать алгоритмы для повышения производительности.

Безопасность

Код не содержит явных уязвимостей, но отсутствие обработки исключений может привести к сбоям. Рекомендуется добавить обработку исключений для повышения надежности.

Обработка ошибок

Обработка ошибок в коде отсутствует. Рекомендуется добавить блоки `try-catch` для обработки возможных исключений, особенно при вводе данных пользователем.

Заключение

Код выполняет свои функции, но требует улучшений в области обработки ошибок, читаемости и структуры. Рекомендуется внести изменения для повышения надежности и удобства использования.